TÓM LƯỢC BÀI GIẢNG

(Vũ Quốc Hoàng)

DANH SÁCH LIÊN KẾT

Chủ đề

- Danh sách
- Danh sách liên kết

Tài liệu

[1] Paul Deiteil, Harvey Deiteil, *C how to program*, Pearson, 8th global edition, 2016.

[2] Vũ Quốc Hoàng, *Bí kíp luyện Lập trình C (Quyển 1)*, hBook, 2017.

Đọc tài liệu

- Danh sách liên kết: Mục 12.1, 12.2, 12.3 và 12.4 [1]

Kiến thức

- Xét yêu cầu: nhận một dãy dữ liệu và lưu trữ lại theo thứ tự nhận (để sau đó xử lý offline) với ràng buộc là không biết trước số lượng. Các chiến lược dùng mảng (cả mảng tĩnh lẫn mảng động) đều không hiệu quả. Cách tốt hơn là dùng danh sách liên kết.
- Đoạn mã và hình minh họa 1

```
#include <iostream>
using namespace std;

struct NODE
{
    int data;
    NODE *next;
};

struct LIST
{
    NODE *first;
    NODE *last;
};

void init(LIST &list)
{
    list.first = list.last = NULL;
```

```
}

void addlast(LIST &list, int x)
{
    NODE *node = new NODE;
    node->data = x;
    node->next = NULL;

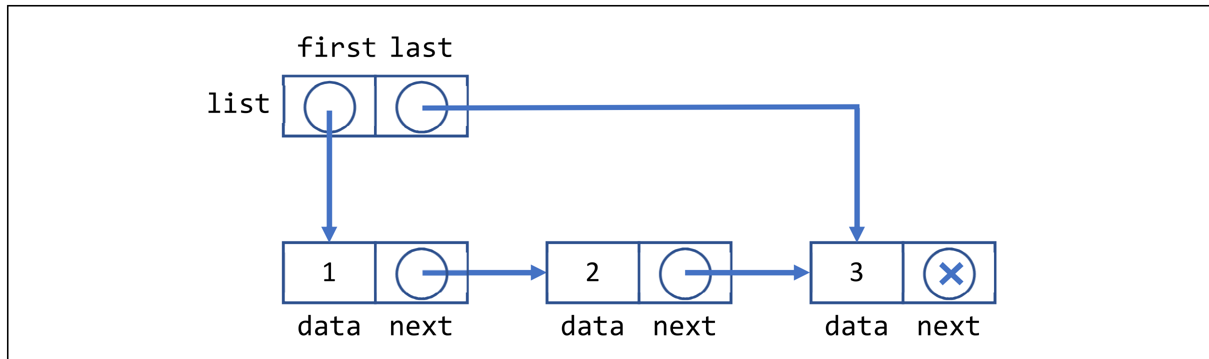
    if(list.last == NULL)
        list.first = list.last = node;
    else
    {
        list.last->next = node;
        list.last = node;
    }
}

void print(LIST list)
{
    for(NODE *p = list.first; p != NULL; p = p->next)
        cout << p->data << " ";
}

int main()
{
    LIST list;
    init(list);

    int x;
    cout << "Nhap cac so nguyen (cho den 0 thi dung):\n";
    cin >> x;
    while(x != 0)
    {
        addlast(list, x);
        cin >> x;
    }

    cout << "Danh sach da nhap la: ";
    print(list);
}
```



- *Danh sách* (list) là một dãy hữu hạn, có thứ tự, các phần tử cùng kiểu. Danh sách (a) có n *phần tử* lần lượt là a_1, a_2, …, a_n được kí hiệu là $(a)_n = [a_1, a_2, …, a_n]$ với a_i được gọi là phần tử thứ i của danh sách. Số lượng phần tử của danh sách, n, được gọi là *chiều dài* (length) của danh sách. *Danh sách rỗng* (empty list) là danh sách có chiều dài 0. Ví dụ: danh sách các số nguyên tố nhỏ hơn 1000 (theo thứ tự tăng dần) là [2, 3, 5, …, 991, 997] (có chiều dài là 168?!), danh sách các ngày trong tuần ([Mon, Tue, …, Sun]) có chiều dài là 7, danh sách sinh viên (theo thứ tự mã số sinh viên), danh sách các kí tự (chuỗi, theo thứ tự viết, chẳng hạn "hello" = [h, e, l, l, o], có chiều dài là 5), …
- Các đặc trưng của danh sách là: các phần tử cùng kiểu, có thứ tự, có thể lặp lại (ở thứ tự khác nhau). Hai danh sách được xem là bằng nhau (như nhau) khi chúng có cùng chiều dài và các phần tử như nhau ở cùng thứ tự. Chẳng hạn "hello" là [h, e, l, l, o] nhưng khác [h, e, l, o] hay [h, o, l, l, e].
- Danh sách có thể được *cài đặt* (hiện thực) bằng mảng (mảng tĩnh/động), bằng *danh sách liên kết* (linked list) hay bằng cách khác. Tùy từng trường hợp cụ thể mà cách cài đặt nào là tốt hơn, hiệu quả hơn. Danh sách liên kết thường hiệu quả hơn mảng trong trường hợp mà số lượng phần tử của danh sách biến động nhiều và không giới hạn (thêm, xóa nhiều trên danh sách).
- Danh sách liên kết là một tập các *nút* (node) mang dữ liệu (là các phần tử của danh sách) liên kết với nhau (theo thứ tự của các phần tử). Trong C/C++, liên kết được hiện thực bằng con trỏ. Một nút, như vậy, ngoài dữ liệu sẽ có con trỏ trỏ đến nút kế tiếp (nút mang phần tử có thứ tự ngay sau). Cấu trúc nút (`NODE` trong đoạn mã trên), như vậy, được gọi là *cấu trúc tự trỏ* (self-referential structure).
- Để quản lý danh sách, ta thường dùng một con trỏ trỏ đến nút đầu của danh sách (nút mang phần tử đầu tiên) gọi là con trỏ first hay head. Trong một số trường hợp ta cũng dùng thêm một con trỏ trỏ đến nút cuối cùng của danh sách (nút mang phần tử cuối cùng) gọi là con trỏ last hay tail (như cấu trúc `LIST` trong đoạn mã trên).
- Con trỏ `NULL` (thực chất là hằng 0 được `#define` sẵn) được dùng với ý là “không trỏ” hay không “không có liên kết”. Chẳng hạn, nút cuối cùng của danh sách sẽ không có nút sau nên không có liên kết đến nút sau (thành phần `next` của nút là `NULL` trong đoạn mã trên). Hay khi danh sách rỗng (không có phần tử) thì con trỏ đến nút đầu `first` (và con trỏ đến

nút cuối, `last`) sẽ là `NULL`. Hàm `init` ở đoạn mã trên, như vậy, sẽ khởi tạo danh sách rỗng.

- Để truy cập các phần tử của danh sách, ta xuất phát từ nút đầu (trỏ bởi con trỏ `first`) và đi theo các liên kết (trỏ bởi thành phần `next` của nút) đến hết danh sách (nhận biết bằng giá trị `NULL` trong thành phần `next` của nút cuối). Một khuôn mẫu như vậy được cho trong hàm `print` dùng để xuất các phần tử của danh sách như trong đoạn mã trên.
- Để thêm (hay xóa) một phần tử vào (hay ra khỏi) danh sách, ta cấp phát động (hay giải phóng) cấu trúc nút, đặt dữ liệu (phần tử) vào nút và điều chỉnh các liên kết cho phù hợp. Xem ví dụ về việc thêm một phần tử vào cuối danh sách trong hàm `addlast` trên.
- Về mặt kĩ thuật, các nút của danh sách liên kết không cần phải được cấp phát kề nhau trong bộ nhớ (thứ tự được quản lý tường minh bởi các liên kết), còn trong mảng thì các phần tử phải được cấp phát kề nhau trong bộ nhớ (thứ tự được quản lý ngầm định bằng chỉ số), nên danh sách liên kết mang lại khả năng tổ chức linh động hơn mảng.
- Đoạn mã và hình minh họa 2

```cpp
#include <iostream>
using namespace std;

struct NODE
{
    int data;
    NODE *next;
};

void insert(NODE *&head, int x)
{
    NODE *node = new NODE;
    node->data = x;

    NODE *p1 = NULL;
    NODE *p2 = head;
    while(p2 != NULL && p2->data <= x)
    {
        p1 = p2;
        p2 = p2->next;
    }
    node->next = p2;

    if(p1 == NULL)
        head = node;
    else
        p1->next = node;
}

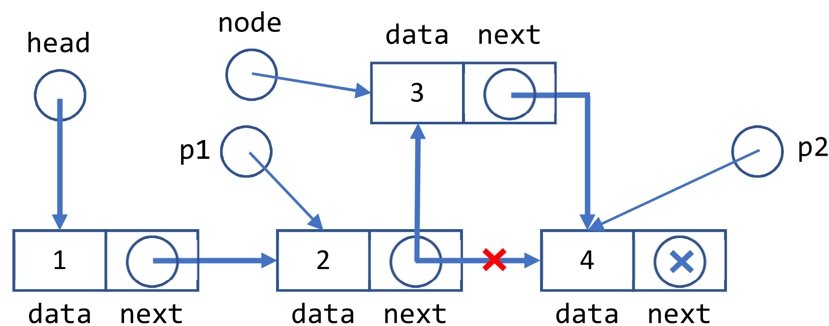
void print(NODE *head)
{
```

```
    for(NODE *p = head; p != NULL; p = p->next)
        cout << p->data << " ";
}

int main()
{
    NODE *head = NULL;

    int x;
    cout << "Nhap cac so nguyen (cho den 0 thi dung):\n";
    cin >> x;
    while(x != 0)
    {
        insert(head, x);
        cin >> x;
    }

    cout << "Danh sach da nhap theo thu tu tang dan la: ";
    print(head);
}
```



- Chương trình minh họa trên cho nhập một danh sách số nguyên (không biết trước chiều dài) và lưu trữ lại theo thứ tự tăng dần của các số. Danh sách liên kết cực kì phù hợp cho việc xử lý “online-offline” này. Nghiên cứu hàm `insert` để thấy cách điều chỉnh các liên kết để chèn thêm một phần tử vào đúng vị trí (theo thứ tự tăng dần của các phần tử).

Kĩ năng

- Biết cách khai báo và thao tác trên danh sách liên kết
- Biết cách vận dụng danh sách liên kết để tổ chức và xử lý dữ liệu

Lưu ý

- Danh sách liên kết là một cấu trúc dữ liệu rất quan trọng nên sinh viên cần rèn luyện nhiều để quen thuộc và thành thạo

Bài tập

Làm lại các bài tập về mảng bằng danh sách liên kết